Redes Neuronales y Aprendizaje profundo

Tema 2 – Redes Neuronales Convolucionales

Irina Arévalo

Contenido

1. Procesamiento de imágenes
2. Convolución
3. Redes convolucionales
4. Arquitecturas preentrenadas

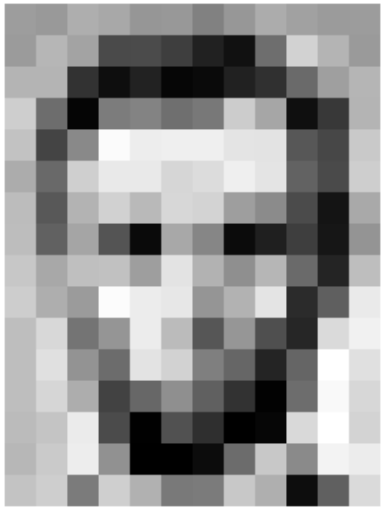
1. Tipos de datos

- Ya hemos trabajado con datos estructurados en forma de tabla que internamente en Python se transforman en arrays de dos dimensiones

Name	Pclass	Sex	Age	Siblings/Spouses Aboard	Parents/Children Aboard	Fare	Survived
Mr. Owen Harris Braund	3	male	22	1	0	7.25	0
Mrs. John Bradley (Florence Briggs Thayer) Cumings	1	female	38	1	0	71.2833	1
Miss. Laina Heikkinen	3	female	26	0	0	7.925	1
Mrs. Jacques Heath (Lily May Peel) Futrelle	1	female	35	1	0	53.1	1
Mr. William Henry Allen	3	male	35	0	0	8.05	0
Mr. James Moran	3	male	27	0	0	8.4583	0

1. Imágenes

- Las imágenes en blanco y negro se representan como una matriz pxq, donde p y q son el número de píxeles por fila y columna respectivamente, y por una cantidad entre 0 y 255, donde 0 representa el negro puro y 255 representa el blanco puro

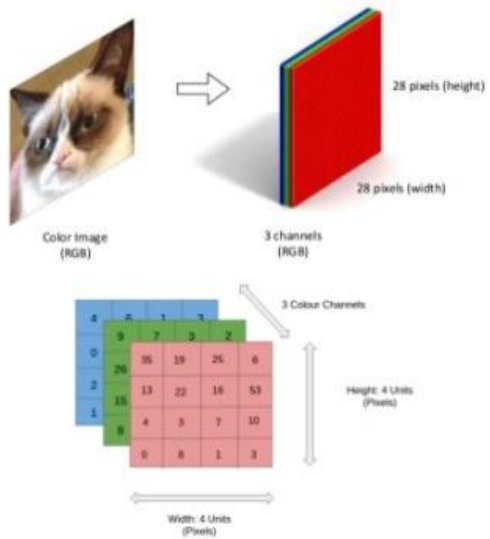


157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

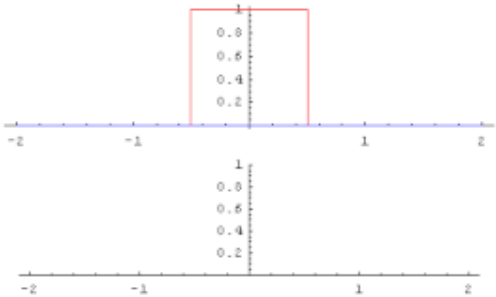
1. Imágenes

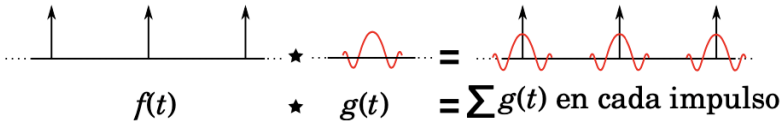
- En general, la imagen i-ésima es un elemento $x_i \in [0, 255]^{H \times W \times C}$, donde H × W es la resolución de la imagen (altura × anchura) y C es el número de canales (1 si es en escala de grises, 3 si es RGB, etc.)
- Por tanto, un conjunto de N imágenes será un tensor $X \in [0,255]^{N \times H \times W \times C}$

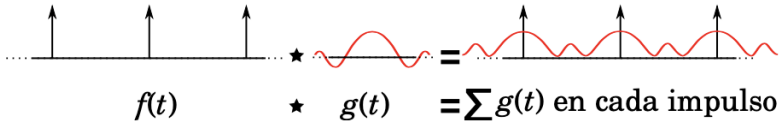


2. Convolución

- Una convolución es un operador matemático que transforma dos funciones en una tercera función que representa una media móvil generalizada.
- La versión discreta es $f[n] \star g[n] = \sum_{k=-\infty}^{+\infty} f[k] \cdot g[n-k]$

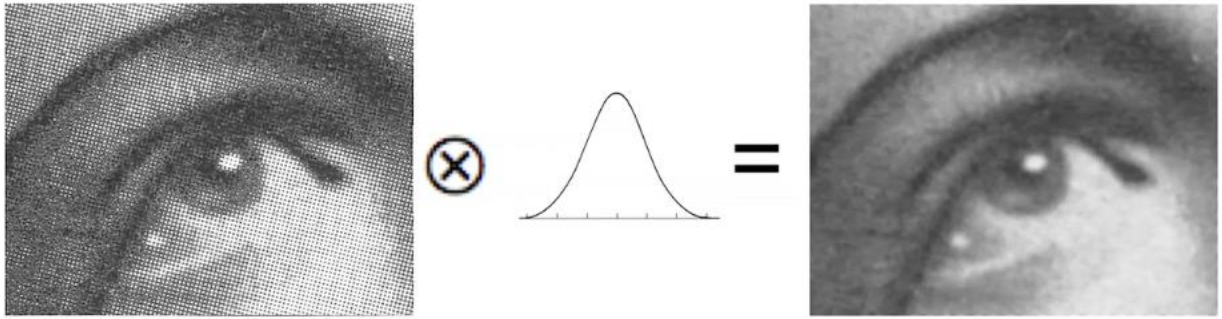






2. Convolución en imágenes

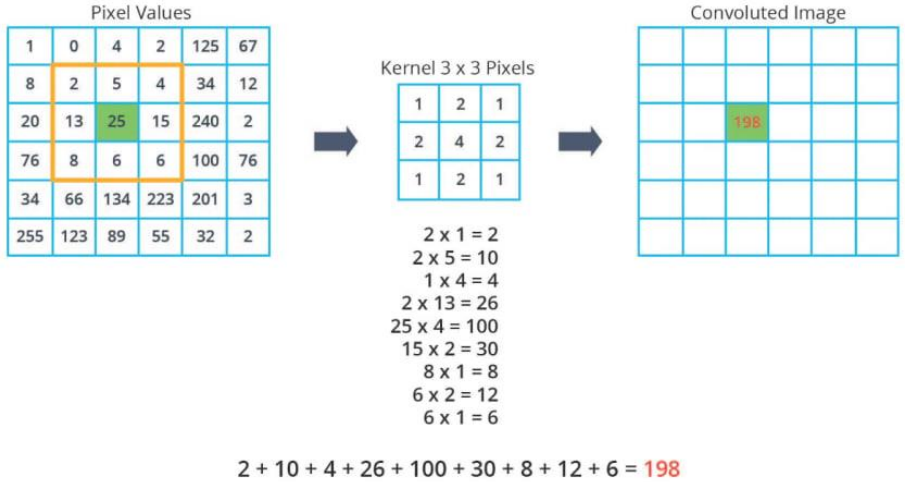
- Tomando una imagen como un array podemos convolucionar esa matriz por otra matriz
- Por ejemplo, al convolucionar una imagen por una distribución normal bivariada, la campana se clonara en cada pixel escalada según el valor de intensidad de dicho pixel, y como la campana es más ancha que la distancia entre pixels los clones se solaparán sumándose, por lo que la imagen resultante es la imagen inicial suavizada



2. Convolución en imágenes

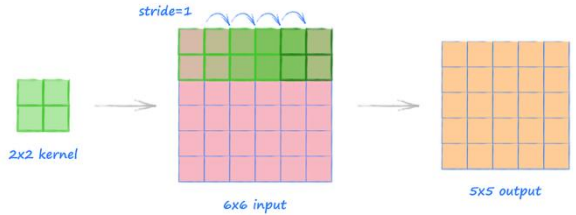
- Por lo tanto convolucionar una imagen por un filtro o kernel consiste en:

1. Elegir una ventana del mismo tamaño que el filtro
2. Convolucionarlos, obteniendo un píxel de la imagen resultante 3. Desplazar la ventana en horizontal, vertical o ambos
4. Repetir desde 2 hasta que se recorra toda la imagen.

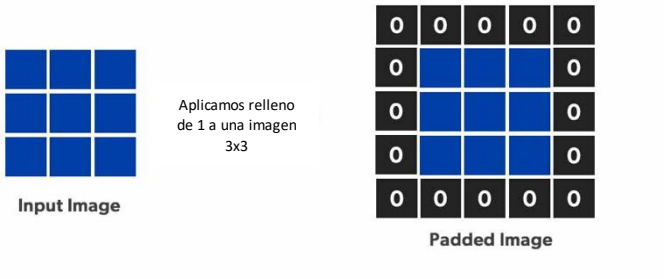


2. Convolución en imágenes

- Los parámetros de la convolución son:
  - **Size** (tamaño): tamaño del filtro
  - **Stride** (paso): número de píxeles que avanza el filtro al hacer la convolución



  - **Padding** (relleno): para evitar problemas en los bordes de la imagen, creamos un borde de ceros



2. Convolución en imágenes

Combinando estos tres parámetros podemos controlar el tamaño de la imagen resultante. Sea O la anchura (altura) de la imagen de salida, I la anchura (altura) de la imagen de entrada, K el tamaño del nucleo (cuadrado), P el valor de padding, y S el valor del stride horizontal (o vertical), entonces

$$O = 1 + \frac{2P + I - K}{S}$$

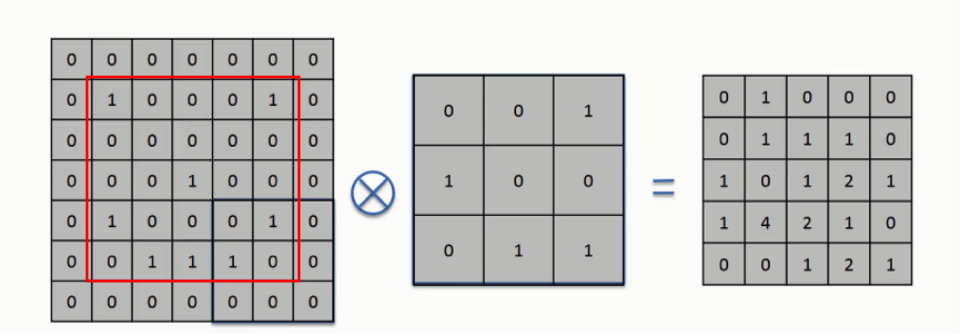


Imagen 5x5, Padding = 1, Stride = 1, Kernel = 3, entonces O=1+ (2*1+5-3)/1 = 5

2. Convolución en imágenes

Name	Kernel	Image Result
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Mean Blur	$\begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}$	

Laplacian	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
Gaussian Blur	$\begin{bmatrix} 1/16 & 2/16 & 1/16 \\ 2/16 & 4/16 & 2/16 \\ 1/16 & 2/16 & 1/16 \end{bmatrix}$	

2. Convolución en imágenes

- La convolución va a ayudar a la red neuronal a aprender características de las imágenes porque:
  - Consigue que cada neurona esté conectada a una pequeña región de la imagen (conectividad local)
  - La convolución crea mapas de características, que son capaces de detectar esquinas, aristas… imitando cómo los humanos ven las imágenes
  - La arquitectura jerárquica hace que las capas ocultas aprendan nuevas características (aristas en las capas iniciales, formas en las capas intermedias, detalles en las finales), por lo que las redes de imágenes suelen aprender más cuantas más capas tengan
  - Reduce el sobre entrenamiento, un problema habitual en las redes
  - Mejora la eficiencia del procesamiento de imágenes

3. Redes Convolucionales

En 2012 Alex Krizhevsky, en el equipo de Geoffrey Hinton, presentó la red AlexNet, basada en redes convolucionales, al ImageNet Large Scale Visual Recognition Challenge, una competición de reconocimiento de imágenes. Su red arrasó al resto de competidores, bajando enormemente el error respecto a retos pasados. Este hito se considera habitualmente el punto de arranque de la revolución del Deep Learning.

3. Redes Convolucionales

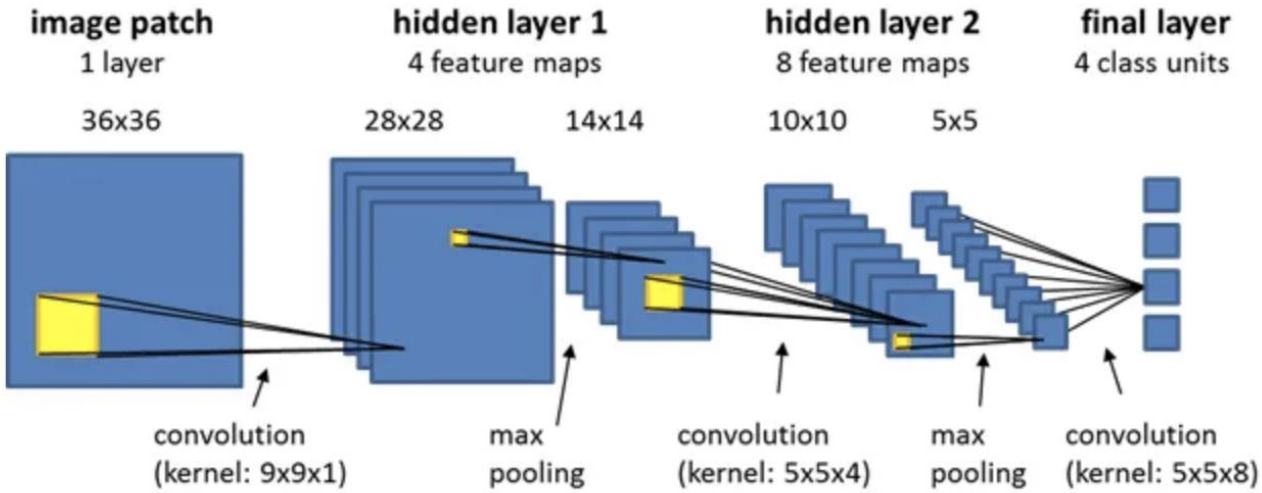
- Una red CNN típicamente empieza por una etapa convolucional que recibe la imagen y tiene la función de extraer características visuales de ella.
- Las neuronas involucradas son filtros convolucionales que se aprenden en el proceso de entrenamiento. Un filtro reaccionará a una sola característica visual, por tanto si queremos aprender varias, debemos usar varios filtros.
- Las características visuales que es capaz de aprender un filtro también dependen del tamaño del filtro. Uno pequeño, 3 × 3 por ejemplo, será capaz de resaltar bordes horizontales, verticales, ... Sin embargo, al aumentar el tamaño el número de posibilidades explota.
- La solución propuesta y utilizada habitualmente consiste en crear una jerarquía de características visuales, organizada en capas

3. Redes Convolucionales

- El proceso para extraer las características de las imágenes es como sigue:

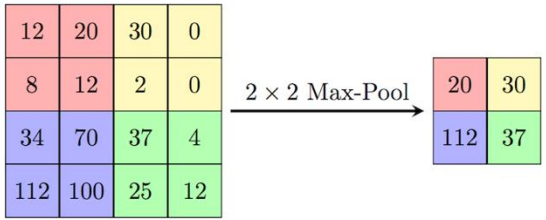
1. Se pasa la imagen por una capa de convolución para extraer características visuales de bajo nivel
2. La salida resultante de convolucionar la imagen con el filtro debe ser modulada por una función de activación
3. Esta salida se reduce en altura y anchura, dando una imagen en la que las características se han aproximado localmente entre sí (capa de *pool)*
4. Se pasa por una nueva capa de convolución que extraiga las características visuales de la salida anterior. Como en la capa 2 sólo han sobrevivido los píxeles de mayor valor, las características visuales que se aprenden en esta etapa son, en cierta medida, la unión espacial características del nivel anterior, más bajo
5. La salida anterior es modulada por una función de activación
6. La salida anterior se reduce en altura y anchura, aproximando las características de nuevo
7. El proceso se repite hasta que sea posible, o hasta que se considere oportuno. Por ejemplo con una imagen de resolución 4096 × 4096, tras 10 repeticiones, se llega a una imagen de resolución 4 × 4, en la que ya casi no podemos distinguir que hay representado.

3. Redes Convolucionales

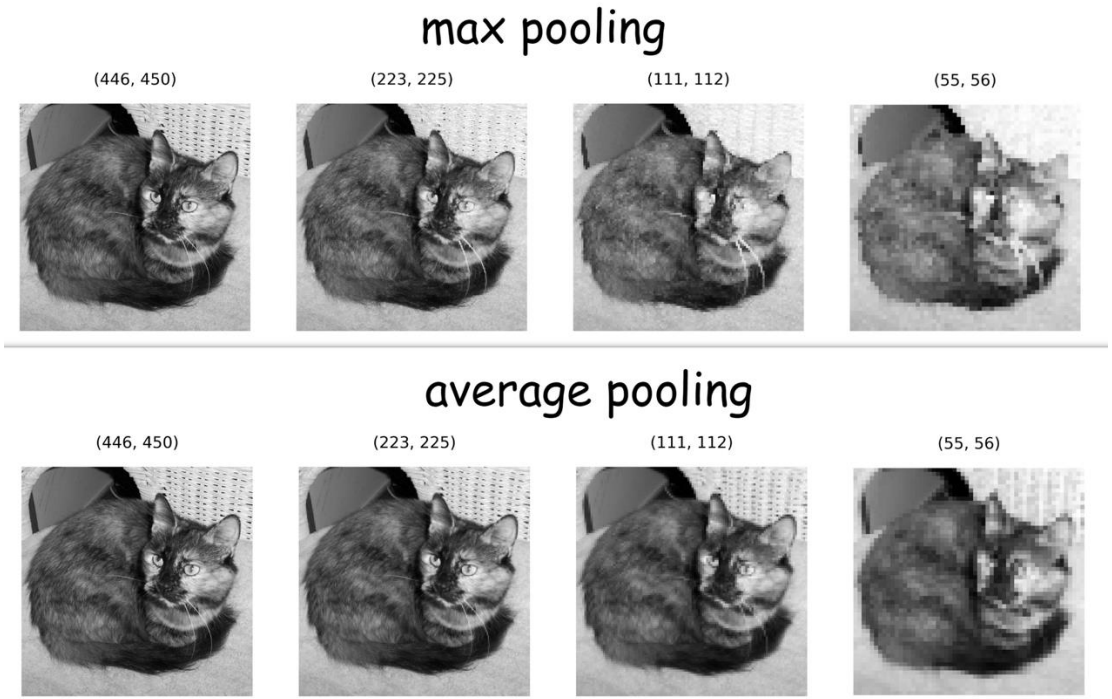


3. Redes Convolucionales

- Al igual que con las capas densas, la función de activación más habitual es la ReLU. Esta función hace que no todas las neuronas se activen siempre, sino sólo aquellas que producen una respuesta positiva y tiene una mejor propagación del gradiente hacia atrás ya que su derivada respecto de x es 0 ó 1
- La agrupación espacial de características se realiza en la capa de pooling. El pooling tiene los mismos parámetros, con el mismo significado, que la operación de convolución. Es decir, la vecindad se va moviendo sobre la imagen, lo mismo que hacía el filtro en la convolución, sólo que ahora la operación es diferente.
- Las dos operaciones más habituales de pooling son max pooling, en el que la vecindad se reduce al máximo valor, y average pooling, en el que se hace la media de la vecindad.

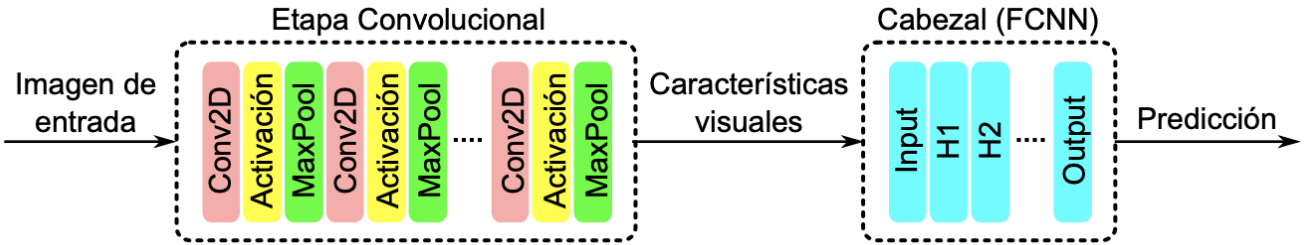


3. Redes Convolucionales



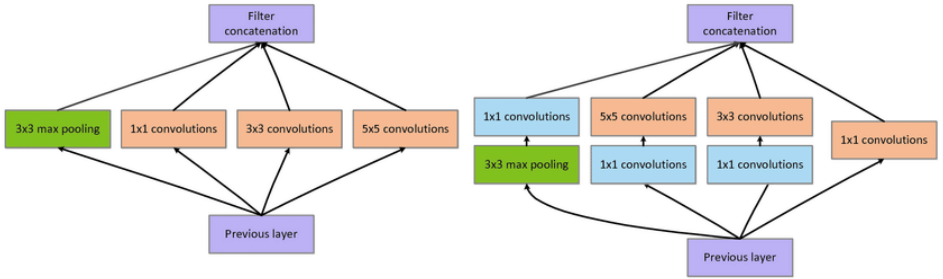
3. Redes Convolucionales

- La etapa convolucional que hemos explicado realiza una extracción de características visuales de la imagen de entrada.
- De esta manera, podemos sustituir la imagen original por sus características visuales y utilizar una red neuronal densa para realizar tareas de clasificación o regresión.
- La ventaja de esta arquitectura es que tanto la etapa convolucional como la red densa son neuronas que se pueden entrenar con back propagation. Es decir que, si por ejemplo estamos aprendiendo a clasificar imágenes, el error de clasificación que se comete con una imagen sirve tanto para actualizar los pesos del cabezal como para aprender mejor las características visuales de las imágenes.



4. Arquitecturas preentrenadas

- Las arquitecturas que involucran redes convolucionales pueden ser muy extensas y pesadas. Por eso distintos grupos de investigación han publicado en abierto sus redes súper densas para que cualquiera pueda reentrenarlas con sus datos y aplicarlas a su caso de uso con una fracción del coste de computación.
- Uno de los primeros ejemplos es **Inception**, de Google. Estos investigadores crearon un módulo Inception que realiza en paralelo la convolución con filtros de diferentes tamaños, además de una reducción de la misma con Pooling, para luego concatenar los resultados.



(a) Inception module, initial form

(b) Inception module with dimension reductions

4. Arquitecturas preentrenadas

- Otro ejemplo es **ResNet**, en el que modifican las transformaciones tras las convoluciones para evitar el desvanecimiento del gradiente (la derivada de la función de coste respecto a los pesos cada vez es más pequeña y la red deja de aprender con el descenso del gradiente, problema muy común en redes con muchas capas)
- Estas arquitecturas de tantas capas normalmente necesitan de un conjunto de entrenamiento enorme y de muchísimo tiempo de ejecución. En la práctica, podemos aplicar estas redes para nuestro problema, cambiando la capa de salida para nuestro problema de regresión o clasificación. Después se entrena la red con el nuevo conjunto de datos, pero actualizando sólo las últimas capas de la misma. La intuición es que las características visuales de bajo nivel serán comunes para cualquier clase de imágenes, por lo que sólo necesitamos aprender las características de alto nivel y como se conectan entre ellas para predecir la clase

Tema 2 – Redes Neuronales Convolucionales

Ingeniería Matemática

Profesora Dra. Irina Arévalo

Curso 2024-25